

Narrative Situation Assessment for Human-Robot Interaction

Björn Jensen, Roland Philippsen, Roland Siegwart

Autonomous Systems Lab, EPFL-I²S-ASL
Swiss Federal Institute of Technology Lausanne (EPFL)
{bjoern.jensen, roland.philippsen, roland.siegwart}@epfl.ch

Abstract—In this paper we address the problem of interpreting sensory data for human-robot interaction, especially when gathered from several robots at the same time. After describing motion tracking in this context, we introduce a general framework for situation representation, and how it simplifies extraction of information suitable for complex man-machine dialogs. As a concrete implementation thereof, a narrative description of a complex scene in a public exposition is created. We regard issues of interpreting sensor data in an efficient way and discuss the effects of the number of robots on the results of the scene interpretation to show that our approach is not only scalable but also profits from a growing number of robots.

I. INTRODUCTION

Research in human-robot interaction has identified awareness as one of the key elements for complex interaction. By knowing where humans are in the vicinity of the robot one hopes to ease the man-machine communication.

In this context we address the question how the information gained in the process of tracking dynamic objects can be used for communication. Several systems for tracking persons with a mobile robot have been presented so far [11], [9], [8]. Static laser sensors were used in [5] to track people. Situation recognition in the context of navigation has been presented in [7]. The question of fusing information in a decentralized network is addressed in [4]. The novelty of our approach is that we are able to convert abstract spatial data gathered from sensors on different mobile robot platforms into textual information usable for man-machine interaction, a process we call *narrative situation assessment*. Using a multi-robot system we were interested in a scalable approach, exploiting when possible the larger perceptive scope of several robots.

Our approach is based on laser range data. Compared to visual information this data lacks color and textual information. The fact that it provides distance information with high precision eases the integration of measurements in the estimation of object's state and tracking.

The results presented herein are obtained from data of highly dynamic situations with often up to a hundred persons in the same space with the robots. The data was collected during the Swiss National Exposition, where ten autonomous mobile robots were interacting during a period of five month with more the 600'000 visitors.

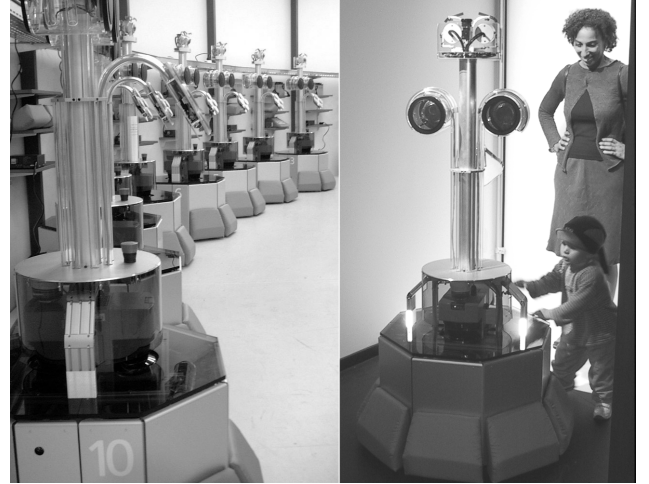


Fig. 1. On the left side: the RoboX team at the charging station during the Expo.02 event. On the right side: one RoboX interacting with visitors and taking a photo of them.

II. ROBOX

RoboX has been developed for human-robot interaction in a public mass exposition. It uses a mechanical face with seven degrees of freedom as an anchor of communication, speech synthesis in four languages and face tracking to enable even users with little or no experience in mobile robotics to interact with it. The design of the robot uses common features for communication, situating its appearance somewhere between anthropomorphic and machine.

For good visibility even in crowded environments we constructed RoboX (figure 1) to be of approximately average visitor's height. Basically, the robot consists of a mobile base with an interactive top, making the face easy to look at, as explained in [6]. Two differential-drive wheels located at the center of the robot allow on the spot turns. Two castor wheels, one at its back and one, with a suspension at its front, ensure the stability of the mobile base. Obstacle avoidance and reliable localization [1] ensure that the robot knows at all times its position and does not collide with visitors or parts of the exposition. As an additional means of security, touch sensitive plates and foam bumpers ensure that the robot stops if running

into anything. Two SICK Laser scanners mounted at knee height provide environmental information for navigation and interaction. A camera mounted in one of the robot's eyes provides additional information for the interaction. Furthermore, the mobile base houses motor controllers, batteries for 10h autonomy, a PowerPC 750 clocked at 400 MHz dedicated for navigation and obstacle avoidance and a Pentium III running at 700 MHz, 128 MB RAM on Windows 2000 for all interaction tasks. Both computers can communicate with each other over a 10 Mbit/sec local Ethernet and with a central computer over wireless interfaces to allow monitoring the state of the robot for security reasons [12].

III. MOTION DETECTION

Motion detection means comparing data acquired at different time instants and identifying the parts that changed. When detecting motion around a mobile robot several problems surface.

- Displacement: the robot's movement implies that data acquired at $t-1$ and t is not in the same coordinate system. A common coordinate system can be established by compensating the robot's motion. This requires precise localization information.
- Dynamic occlusion: motion in the vicinity of robot will hide previously visible parts of the environment and uncover parts which were hidden before. In order to correctly identify motion we distinguish this from static occlusion.
- Static occlusion: when moving, the robot will uncover regions it could not see beforehand. This occurs even in a static environment and has to be treated differently from motion.

To find motion with a laser range finder, we propose a method based on so-called *polar motion arrays* of range readings. We maintain three such arrays: The current readings, the previous readings, and the accumulated knowledge of immobile elements (called static map). These three arrays are used as follows to detect motions (see figure 2):

- 1) Transform the static map to the new robot position. Due to the concrete nature of the array, some of its fields can become unoccupied and are set to ∞ , which is always replaced by the next available reading.
- 2) Compare the current readings with the static map. Differing readings become candidate motions, while similar readings overwrite the old values in the static map to fill it up again and avoid drift. Comparing them with a static map solves the problem of dynamic occlusions.
- 3) The motion candidates are then compared with the previous scan by checking whether they could have been sensed before. Only candidates that would have been visible before correspond to real motions, the

others are due to static occlusion and thus are also used to update the static map.

We denote the distance measured at a certain angle ϕ_i as r_i , where $\phi_i = 2\pi i/N$ with $i = 0 \dots N-1$ and N the number of readings per scan, assuming that our laser sensors are mounted on the center of the robot's coordinate system. Indexing the array containing the current scan is straightforward, as N is constant: It is filled in the order with which readings are read from the sensor.

The localization information needed for the transformations is denoted as x_t and y_t for the position and Θ_t for the orientation. Equation 1 shows how to transform readings to the current pose. The inverse transformation is obtained by swapping $t-1$ and t .

$$\begin{aligned} dx &= x_t - x_{t-1} \\ dy &= y_t - y_{t-1} \\ a &= r_{t-1} \cos(\phi_{t-1} + \Theta_{t-1}) - dx \\ b &= r_{t-1} \sin(\phi_{t-1} + \Theta_{t-1}) - dy \\ r_t &= \sqrt{a^2 + b^2} \\ \phi_t &= \arctan(b/a) - \Theta_t \end{aligned} \quad (1)$$

The advantage of the polar motion array is the ease of comparisons between readings. For each pair of elements from the current and the static arrays, we use equation ?? to determine motion candidates. To filter static occlusions, we use equation 2.

$$\begin{aligned} r_{t,i} < r_{\text{static},i} - \Delta_i &\Rightarrow i \in \{C\} \\ \Delta &= \min(\max(\frac{|r_{\text{static},i-1} - r_{\text{static},i+1}|}{2}, \Delta_{\min}), \Delta_{\max}) \\ i \in \{C\} \cap r_{t,i}^* > r_{t-1,i} &\Rightarrow i \in \{S\} \end{aligned} \quad (2)$$

where $\{C\}$ is the set of all motion candidates, $\{S\}$ is the entity of all static occlusions, and $\Delta_{\min} = 0.03m$, $\Delta_{\max} = 0.1m$ are parameters, mainly depending on the precision of information on the robot's position and the quality of the sensor readings available.

After the calculations described above, motions are known on a per-row-data basis. In order to treat the environment as composed of objects, the dynamic elements are clustered and their center of gravity is computed and used to represent the object in the following.

IV. TRACKING DYNAMIC OBJECTS

To follow humans through the exposition, we need to establish a relation between motion sensed at different instants of time. We use a Kalman Filter based tracking scheme that links motion elements minimizing the Mahalanobis distance to the predicted position. In the following we use a similar nomenclature as [2], [3].

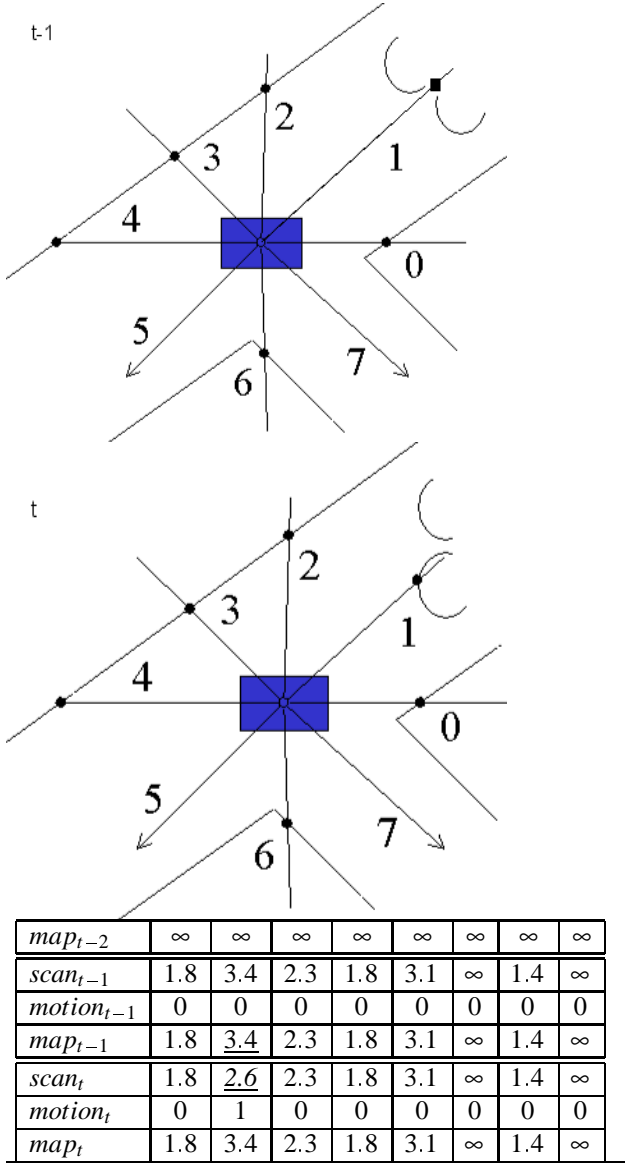


Fig. 2. An example of environment as seen from the robot at two different instants of time. Below the table shows how the scans relate to the static map and how motion is detected.

Single Motion Tracking: In our current implementation, we use a state vector $\vec{x} = (x, \dot{x}, y, \dot{y})^T$ and a constant velocity model to predict the object's motion as shown in equations 3 (transformation matrix) and 4 (process noise matrix). With T as the timestep of the algorithm.

The relation of the state vector \vec{x} to the observation \vec{z} is given by the observation matrix H in equation 5.

The process of tracking can be divided into initialization and update of the tracker. For initialization we use the center of gravity found by the motion detection. Since velocity is unknown at first, we assign zero to it, albeit with a large uncertainty (which will decrease in the subsequent steps as the tracker continues to match). Matching

is done minimizing the Mahalanobis distance between $(\vec{m}_j - \vec{z}(t+1|t))S^{-1}(\vec{m}_j - \vec{z}(t+1|t))^T$ over all available motion elements, where \vec{m}_j is the motion, \vec{z} is the predicted observation, and S is the innovation of observation. In order to avoid arbitrary matches we allow only those elements to link that are below a threshold obtained from the χ^2 -distribution.

$$F = \begin{pmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (3)$$

$$Q = q \begin{pmatrix} T^3/3 & T^2/2 & 0 & 0 \\ T^2/2 & T & 0 & 0 \\ 0 & 0 & T^3/3 & T^2/2 \\ 0 & 0 & T^2/2 & T \end{pmatrix} \quad (4)$$

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$\vec{z} = H\vec{x} \quad (5)$$

Tracking Multiple Objects: Our aim is to track several objects at the same time. Using multiple Kalman filters naively may result in tracking the same object with several filters, which is not desirable. Computing the Mahalanobis distances separately from the matching step helps avoiding this problem. Thus we obtain a matrix G with the number of trackers as rows and the number of detected motions as columns (see equation 6).

$$G = [g_{ij}]$$

$$g_{ij} = (\vec{m}_j - \vec{z}_i(t+1|t))S_i^{-1}(\vec{m}_j - \vec{z}_i(t+1|t))^T \quad (6)$$

where \vec{z}_i is the predicted observation, and S_i is the innovation of observation for the tracker i .

In the matching step, each tracker is assigned the motion minimizing its Mahalanobis distance. Any resulting conflicts are resolved as follows:

- 1) For a given motion with more than one tracker, assign a cost, which is the difference between its two smallest Mahalanobis distances to each tracker and choose the one with the lowest cost.
- 2) The now unassigned trackers are assigned to the motion corresponding to their second-to-smallest Mahalanobis distance.
- 3) Iterate the two previous steps until a one-to-one mapping between motions and trackers is achieved.

As cost of a tracker we use the difference between its two smallest Mahalanobis distances. If necessary, this simple but robust procedure could be replaced by the Vogel algorithm, which minimizes the global cost of the assignment problem. A more general solution to the

data association problem of tracking multiple objects is the multiple hypothesis tracking [10] with the known limitations for real-time applications. A completely decentralized data fusion scheme can be found in [4].

V. NARRATIVE SITUATION ASSESSMENT

By situation assessment, we refer to the act of detecting those states and events in the object-relationships which are useful for human-robot interaction. For instance, a robot might inform a visitor that she just passed an interesting exhibit on her left-hand side, or a robot might start reacting to someone as soon as they come closer than a certain distance. Sharing this information with a community of robots further enhances the interaction, since robots can refer to situation in which they did not even remotely participate.

States are obtained from the combined information of all robots and represented using several matrices. Events are detected by comparing object state vectors (see below) with each other. By object we mean any entity of interest to human-robot interaction — in our case humans, robots, and static exhibits. In order to avoid a view of the environment centered on a special entity, we represent object relationships in various matrices to give us a common framework.

As stated, we assume that the world we are analyzing consists of three kinds of objects. For the sake of simplicity, we further assume that all these elements are circular with a meaningful definition of forward direction. Each of these objects is assigned a state vector \vec{o} and an environment vector \vec{e} , given in equation 7 and illustrated in figure 3.

$$\begin{aligned}\vec{o} &= (ID, t, x, y, v, \Phi)^T \\ \vec{e} &= (r_{\min}, r_{\text{close}}, \alpha_{fr}, \alpha_{fl}, \alpha_{bl}, \alpha_{br})^T\end{aligned}\quad (7)$$

where ID identifies the object, t represents its type, (x, y) its position, v its translational speed, and Φ its heading; r_{\min} can be considered the object radius, r_{close} defines when another object is close to it, and the various $\alpha \in (-\pi, \pi]$ define which regions lie to the front-left, front-right, back-left, and back-right of the object.

The two main matrices used to represent spatial relationships between objects are $R(t)$ for the euclidean distances (eq. 8) and $A(t)$ for the angle between an object's heading and the relative position of the other objects (eq. 10). While $R(t)$ is symmetrical, $A(t)$ can take any form (we set its diagonal to zero).

We define five more matrices, one for each element of \vec{e} to provide unified calculation of events between objects: R_{\min} , R_{close} , A_{fr} , A_{fl} , A_{br} , and A_{bl} . Currently, all these have elements that are identical along a row. We anticipate future developments of our algorithm to take into account

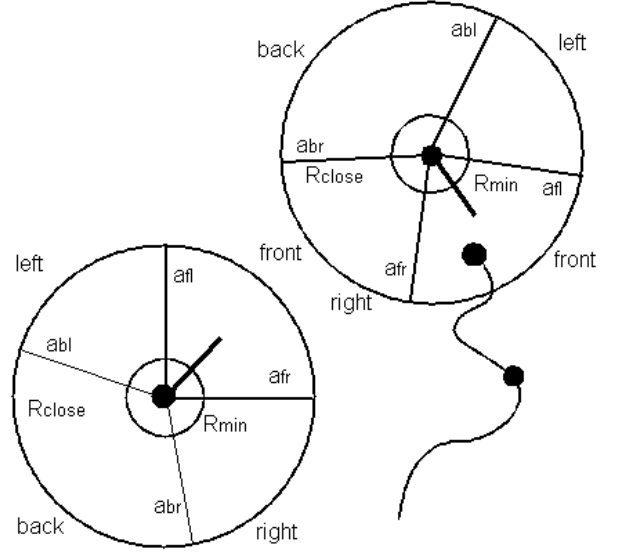


Fig. 3. Definition of an object's vicinity. And an example of event "object approaching".

more details of object relationships and rely on matrices of general form.

To cope with the dynamics of the situation, we must allow the matrices to grow and shrink according to the current number of objects. The relation of the matrix rows to the objects is maintained by a column vector l of the actual object IDs (eq. 10).

$$\begin{aligned}R &= [r_{ij}] \\ r_{ij} &= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}\end{aligned}\quad (8)$$

$$A = [\alpha_{ij}] \quad (9)$$

$$\alpha_{ij} = \begin{cases} 0 & \text{if } i = j \\ \arctan\left(\frac{y_j - y_i}{x_j - x_i}\right) - \Phi_i & \text{otherwise} \end{cases}$$

$$l = (ID_1, ID_2, \dots, ID_n)^T \quad (10)$$

All these matrices serve the purpose of creating a narrative of events in the environment, which we calculate by applying certain rules to the comparisons between the matrices (see section VII). Currently, we use a relatively direct mapping to a narrative text.

We distinguish between static narration, comparing values only at the current time instant, and dynamic narration, which takes into account historical evolution of object state vectors. An example is shown in figure 3.

VI. SCALABLE MULTI-ROBOT SYSTEM

The approach is valid for a single robot and has also been extended to take advantage of multi-robot systems,

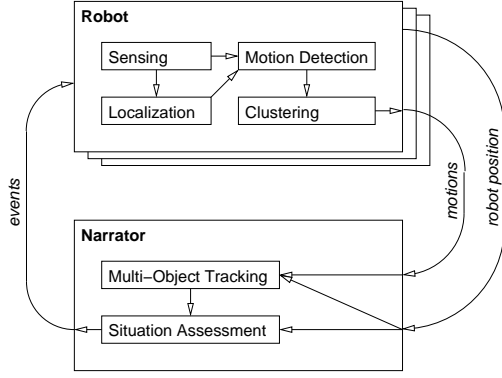


Fig. 4. Structure of the narrative system. Upper part is running on the N robots, lower part on the central server.

as these can effectively cover more area by combining their laser readings. In order to distinguish between local (single-robot) and global aspects of such a fusion, we introduce a central computer, called narrator, to communicate with each robot. See figure 4.

- On-board: Localization, motion detection and clustering are done on the robot, using the on-board sensors and the polar array method.
- Off-board: Multi-object tracking and Situation Assessment are done on the narrator, where data from all robots is more easily fused and the state vectors of all objects updated. It also takes care of removing and adding objects as necessary.

In a certain sense, the central computer is a client polling all robots about their information. But then, it becomes a server handing out information about the scene which is used by the robots to interact with visitors.

Two issues arise in such a distribution: Spatial and temporal synchronization. The spatial problem is solved using global localization based on the same a-priori map on all robots [1]. In addition to the localization information stored in the map, we define object and environment vectors for the static objects in our environment (shared by all robots). Clock synchronization can be done using common techniques used for instance on the Internet.

VII. EXPERIMENTS

As a demonstration implementation, the following rules create a human-comprehensible textual description of the events in the environment using equation 11, 12, 13.

$$C = [c_{ij}] \quad (11)$$

$$c_{ij} = (r_{t,ij} < r_{close,i}) \cap (r_{t-1,ij} \geq r_{close,i})$$

$$B = [b_{ij}] \quad (12)$$

$$b_{ij} = (\alpha_{t,ij} \geq \alpha_{fr,i}) \cap (\alpha_{t,ij} < \alpha_{fl,i}) \cap (r_{t,ij} < r_{close,i})$$

$$L = [l_{ij}] \quad (13)$$

$$l_{ij} = (r_{t,ij} \geq r_{close,i}) \cap (r_{t-1,ij} < r_{close,i})$$

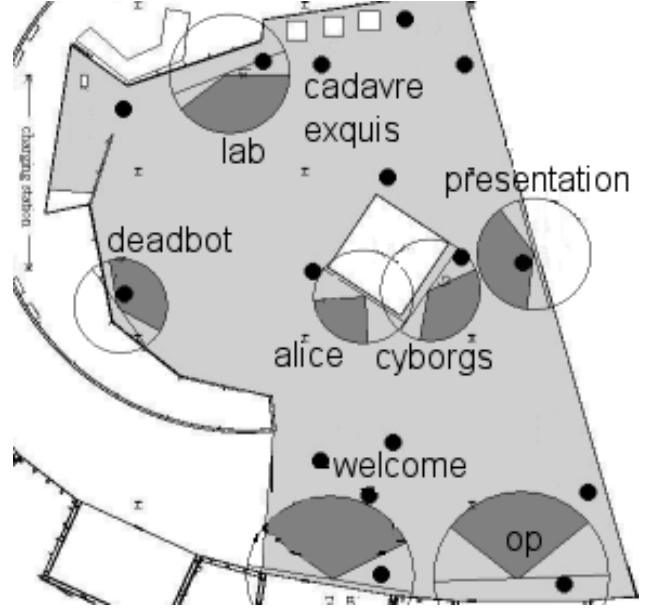


Fig. 5. Experimental setup: the positions of the static object and the interaction zones are shown. Visitors entered the exposition from the lower left corner. The exit is located in the upper right corner.

where c_{ij} , b_{ij} , and l_{ij} define the events of object j is coming close to, is in front of, is leaving object i .

In our experimental setup at the Swiss national exhibition Expo.02, we work with 15 static exhibits (showing for example an industrial robot, the off-road robot Shrimp developed at our lab, and artificial hip joints), up to ten mobile robots and approximately 100 persons on the 315 m^2 exhibition floor.

Since our system is not yet capable of handling so many trackers simultaneously in real-time, we limit it to 20 moving objects (visitors). Initialization of the trackers takes place in a region near the entrance. A tracker is freed and subsequently reassigned if it doesn't match for a predefined number of steps, or if the tracked visitor passes the exit of the exposition. Figure 5 shows the setup of the environment.

In figure 6 you see several tracks of persons through the exposition. Problems that may occur in such a dynamic environment are occlusions and false matches. Occlusions lead to tracks without matches, which are visible as straight lines in 6. Another problem is in matching the wrong candidate, which is mostly due to imperfections of the motion model in describing human motion. Due to the nature of the laser range data these false matches cannot be detected by the tracking algorithm. Incorporation of additional sensor, especially vision, may help to increase robustness. Another point increasing the robustness of the approach is the number of the distributed robots allowing us to monitor most of the exhibition surface and minimizing occlusions. The coverage of the area increases

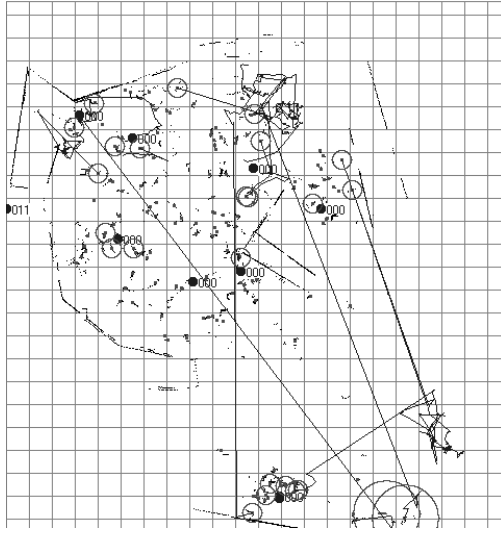


Fig. 6. Several visitors tracked through the exhibition. Big dots indicate the robots' positions, whereas smaller dots show the sensed motion. The tracks are shown as traces.

1	10 : 40 : 37	person 01 entered the exposition
2	10 : 41 : 05	person 01 approached R03 front side
3	10 : 43 : 12	person 01 is coming close to person 12
4	10 : 43 : 18	person 01 is leaving person 12
5	10 : 43 : 47	person 01 arrived at industrial robot
6	10 : 46 : 15	person 01 left industrial robot
7	10 : 48 : 00	person 01 arrived at cyborgs
8	10 : 49 : 05	person 01 left robot 03
9

Fig. 7. The narration generated the track of one visitor. The left column is the event counter, in the middle the time is indicated and on the right side events generated for person 01 are shown.

with every robot added until the point where they start to block each other's field of view. Using the general narrative framework presented herein, we are able to fully exploit this and created an instantaneous narration, figure 7 shows an example for one specific person.

VIII. CONCLUSION

The situation assessment method presented herein provides a useful novel approach to information extraction for human-robot interaction. A first implementation which extracts simple textual narratives demonstrates that it is appropriate, yet the representational power remains relatively unexploited. Ongoing research at the Autonomous Systems Lab is aimed at using this approach to its fullest, particularly for more complex human-robot interaction and group modeling (identify ties among several objects from their motion patterns).

Other challenges are real-time aspects and the eventual use of multi-sensor tracking methods to decrease the number of false matches.

IX. REFERENCES

- [1] Kai-Oliver Arras, Roland Philippsen, Marc de Battista, Martin Schilt, and Roland Siegwart. A navigation framework for multiple mobile robots and its application at the expo.02 exposition. *Workshop: Robots in Exhibitions, IEEE/RSJ IROS*, 2002.
- [2] Yaakov Bar-Shalom and Thomas E. Fortmann. *Tracking and data association*. Academic Press Inc., 24-28 Oval Road, London, 1988.
- [3] Yaakov Bar-Shalom and Xiao-Rong Li. *Estimation and Tracking: Principles, Techniques and Software*. Artech House Inc., 685 Canton Street, Norwood, MA 02062, 1993.
- [4] Hugh Durrant-Whyte and Mike Stevens. Data fusion in decentralised sensing networks. *4th International Conference on Information Fusion*, 2001.
- [5] Ajo Fod, Andrew Howard, and Maja Mataric. Laser-based people tracking. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA-02)*, pages 3024–3029, 2002.
- [6] Björn Jensen, Gilles Froidevaux, Xavier Greppin, Antoine Lorotte, Laetitia Mayor, Mathieu Meisser, Guy Ramel, and Roland Siegwart. The interactive autonomous mobile system roblox. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-02)*, 2002.
- [7] Boris Kluge, Jörg Illmann, and Erwin Prassler. Situation assessment in crowded public environments. *Proc. of Int. Conf. on Field and Service Robotics*, 2001.
- [8] Boris Kluge, Christian Köhler, and Erwin Prassler. Fast and robust tracking of multiple moving objects with a laser range finder. *Proc. of Int. Conf. on Robotics and Automation*, pages 1683–1688, 2001.
- [9] Erwin Prassler, Jens Scholz, and Alberto Elfes. Tracking multiple moving objects for real-time robot navigation. *Autonomous Robots*, 8:105–116, 2000.
- [10] Donald Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, December 1979.
- [11] Dirk Schulz, Wolfram Burgard, Dieter Fox, and Armin Cremers. Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. *Proc. of the 2001 IEEE Int. Conference on Robotics and Automation*, 2001.
- [12] Nicola Tomatis, Grégoire Térien, Ralph Piguet, Daniel Burnier, Samir Bouabdallah, and Roland Siegwart. Design and system integration for the expo.02 robot. *Workshop: Robots in Exhibitions, IEEE/RSJ IROS*, 2002.